WHAT IS CLAIMED IS:

1. A method of converting a non-object oriented computer environment to a new object oriented computer environment, the method comprising the steps of:

identifying an existing object oriented computer environment;

identifying the non-object oriented computer environment;

defining requirements for the new object oriented computer environment;

selecting grammar and syntax compatible with the non-object oriented computer environment;

developing object oriented extensions, wherein an existing application of the non-object oriented computer environment remains executable and wherein the new object oriented computer environment accesses information of the non-object oriented computer environment; and

preparing the new object oriented computer environment, wherein the new object oriented computer environment includes the requirements, the grammar and syntax and object oriented extensions.

2. The method of claim 1, wherein the step of identifying an existing object oriented computer environment includes identifying a commercially available object oriented computer environment.

3. The method of claim 1, wherein the step of identifying the non-object oriented computer environment includes identifying a legacy non-object oriented computer environment.

4. The method of claim 3, wherein the legacy non-object oriented computer environment includes a user language interface and data structures.

5. The method of claim 3, wherein the legacy non-object oriented computer environment allows multiple users.

6. The method of claim 3, wherein the legacy non-object oriented computer environment includes a distributed environment.

7. The method of claim 1, wherein the non-object oriented computer environment allows simulation modeling.

8. The method of claim 6, wherein the non-object oriented computer environment allows simulation modeling for the analysis of the performance of software executing in an computer system.

9. The method of claim 1, wherein the step of selecting grammar and syntax includes selecting the semantics of the non-object oriented computer environment.

10. The method of claim 1, wherein the step of selecting grammar and syntax includes selecting semantics compatible to the non-object oriented computer environment.

11. The method of claim 1, wherein the step of selecting grammar and syntax includes selecting the semantics of the existing object oriented computer environment.

12. The method of claim 1, wherein the step of developing object oriented extensions includes developing an object header structure and an object data structure.

13. The method of claim 11, wherein the step of developing an object header structure includes developing an object header structure that provides a unified object oriented interface to a user and internal objects.

14. The method of claim 11, wherein the step of developing an object data structure includes developing an object data structure containing a data structure of the non-object oriented computer environment.

15. The method of claim 1 further comprising the step of developing general-purpose utility classes.

16. The method of claim 1, wherein the step of preparing the new object oriented computer environment includes creating new code.

17. The method of claim 1, wherein the step of preparing the new object oriented computer environment includes creating an operating system.

18. The method of claim 1, wherein the new object oriented computer environment includes an object oriented computer language.

19. A computer system for simulation modeling, the computer system comprising:

an object oriented programming language;

application software written in the object oriented programming language, wherein the application software simulates computer systems;

in which the object oriented programming language further comprises:

an application logic function;

data types and scope, wherein the data types and scope include data types and scope of a non-object oriented programming language;

a class for message instancing.

20. The computer system of claim 19 wherein the object oriented programming language further comprises:

client workload models;

server process infrastructure;

database models;

operating system models;

statistics capability;

utility classes; and

garbage collection.

21. The computer system of claim 20 wherein the object oriented programming language is Object Oriented ADN.